ITRI 615

Pfleeger Chapter 3

Whitman & Mattord Chapter 2

Nonmalicious program errors, buffer overflows

- Programmers often make mistakes, most of which are unintentional and nonmalicious, even though they can still cause serious harm.
- The first type of error is buffer overflows.
- Occurs when the size of the memory allocated to store data is insufficient.
- The attacker can abuse overflows to replace code in the system space. If the attacker can masquerade as the operating system, they can execute many commands in a very powerful role.
- By causing an overflow in the CPU stack register, the attacker can change the stack pointer to point to a block of code the attacker wants.

Nonmalicious program errors
Incomplete mediation

- Also referred to as unchecked data values. Typically occurs when either an invalid data type is entered into a variable, for example minute is entered into month, or when a value out of range is entered, for example 1800 as the year.
- Typically handled through exception handling, or by testing values as they are entered.
- Through the unnecessary passing of parameters back to the vendor, the users/attackers have the option to alter the total price, which can work if the required security checks are not performed.
- The amount of damage done depends on the amount of time it goes undetected.

Nonmalicious program errors
Time-of-check to time-of-use errors

- Refers to the errors that occur due to the time delay between verifying access rights and giving access.

- For example, a user requests access to a file, a file which access is allowed for. While the access control mediator checks for access clearance, the user could change the file name descriptor to a file for which they do not have access. Thereby exploiting the time delay.

- The main security implication is ineffective access control, thereby seriously comprising security.

- Prevention includes letting the access checking software own the data until the verification has occurred, or to allow no interruption during the validation.

Malicious programs

- Programs only become security threats when data and state changes trigger it.
- Malicious programs can cause anything from mere irritation to serious harm.
- Documented proof of malicious programs can be found as far back as 1972.
- Malicious code is the general name for unanticipated or undesired effects in programs or program parts.

Kinds of malicious code (Table 3.2 p. 170)

- Virus – a program that can replicate itself and pass on malicious code to other nonmalicious programs by modifying them. Can be transient, where its life depends on its host, or resident, where it locates itself in memory where it can remain active or be activated as a stand alone program.

- Trojan horse – Contains unexpected, additional functionality.
- Logic bomb – Triggers action when condition occurs.
- Time bomb – Triggers action when specified time occurs.
- Trapdoor – Allows unauthorized access to functionality.
- Worm – Propagates copies of itself through a network.
- Rabbit – Replicates itself without limit to exhaust resources.

Virus effects and causes (Table 3.4 p. 193)

Virus effect	How it is caused
Attach to executable program	Modify file directory Write to executable program file
Attach to data or control file	Modify directory, rewrite data Append to data, append data to self
Remain in memory	Intercept interrupt Load self in nontransient memory area
Infect disks	Intercept interrupt, intercept operating system call, modify system file or executable program
Conceal self	Intercept system calls Classify self as hidden file
Spread infection	Infect boot sector, systems program, or ordinary program and its data
Prevent deactivation	Activate before deactivating program Store copy to reinfect

Prevention of virus infection

- Several techniques exist for building a reasonably safe community for electronic contact. They include:

1. Use only commercial software acquired from reliable, well established vendors.
2. Test all new software on an isolated computer.
3. Open attachments only when you know them to be safe.
4. Make a recoverable system image and store it safely.
5. Make and retain backup copies of executable system files.
6. Use virus detectors regularly and update them daily.

Virus truths and misconceptions

- Viruses can infect only Microsoft Windows systems. False.
- Viruses can modify hidden or read only files. True.
- Viruses can appear only in data files, Word documents or programs. False.
- Viruses spread only on disks or through e-mail. False.
- Viruses cannot remain in RAM memory after a power off. True.
- Viruses cannot infect hardware. True.
- Viruses can be malevolent, benign or benevolent. True.

Targeted malicious code

- Trapdoors – an undocumented entry point to a module.
- Salami attack – merges bits of seemingly inconsequential data to yield powerful results.
- Rootkit – a piece of malicious code that goes to great lengths not to be discovered or, if discovered and removed, to reestablish itself whenever possible.
- Privilege escalation – a means for malicious code to be launched by a user with lower privileges but run with higher privileges.
- Interface illusions – a spoofing attack in which all or part of a web page is false.

Targeted malicious code *cont.*

- Keystroke logging – retains a surreptitious copy of all keys pressed.
- Man-in-the-middle attacks – a malicious program that interjects itself between two other programs.
- Timing attacks – using processing time as an indication of when the decryption key value is being approached while guessing.

Developmental controls against program threats

- Software development requires people who can

1. Specify the system.
2. Design the system.
3. Implement the system.
4. Test the system.
5. Review the system.
6. Document the system.
7. Manage the system.
8. Maintain the system.

We must design systems that are both secure and usable. You can't retrofit usable security.
Tools aren't solutions. Mind the upper layers. Keep the customers satisfied. Think and act locally.

Modularity, encapsulation and information hiding

- Three key principles of software engineering are:

- Modularity – creating a design or code in small self-contained units, called components or modules.

- Encapsulation – when components are isolated from the effects of other components, fault tracing is simplified and it is also easier to maintain the system.

- Information hiding – by hiding information, each component hides its precise implementation or some other design decision from the others. This simplifies maintenance and aids with system stability.

Other controls against program threats

- Mutual suspicion – where two programs operate as if other routines in the system were malicious or incorrect.
- Confinement – where a program is strictly limited in what system resources it can access.
- Genetic diversity – some people believe that it is risky having many components of a system come from one source.
- Peer reviews – peer reviews in software engineering can be extraordinarily effective in fault identification and correction.
- Hazard analysis – a set of systematic techniques intended to expose potentially hazardous system states.

Other controls against program threats *cont.*

- Testing – a process activity that hones in on product quality, making the product failure free or failure tolerant.
- Good design – use a philosophy of fault tolerance, have a policy for handling failures, capture the design rationale and history, use design patterns.
- Prediction – predict the risks involved in building and using the system, and mitigate these risks.
- Static analysis – locate and repair security flaws in code before a system is up and running.
- Configuration management – know who is making which changes to what and when.

Other controls against program threats *cont.*

- Learn from mistakes – one of the easiest things that can be done to enhance security.
- Proofs of program correctness – involves making initial assertions about the inputs and then checking to see if the desired output is generated.

Important to note that none of the mentioned controls can guarantee the security or quality of a system.

Questions

- Any questions?